Building a Citation Graph for NeurIPS & ICML Papers

Members:
SagarVerma (24DO379)
ShivayVadhera (24D1598)
Mohiboddin (23M0827)
PanavShah (23B3323)

May 4, 2025

Abstract

We construct a citation graph from 6545 NeurIPS and ICML papers (1999–2023). Using Python, `NetworkX`, and parsers for `.bbl / .bib` files, we extract citation pairs, build a directed graph, and compute basic topological statistics. The workflow is encapsulated in the accompanying Jupyter notebook; several expensive cells are commented out after a first run to avoid re-computation. This report summarises the methodology and the key results.

1 Dataset

- **Source:** Curated folders, one per paper, each containing `title.txt`, `abstract.txt`, and bibliography.
- **Size:** 6 575 paper folders.
- **Noise handling:** papers with empty or malformed bibliographies are skipped.

2 Methodology

2.1 Parsing Bibliographies

1. Read each `.bbl / .bib`; extract normalised titles and abstract of cited papers.
2. Map each citation target back to a folder name using the title.
3. Emit directed edges `src` $\rightarrow$ `dst` for all matches.

2.2 Graph Construction & Cleaning

- Built with `networkx.DiGraph`.
- Self-loops removed (`G.remove_edges_from(nx.selfloop_edges(G))`).
- Largest weakly-connected component retained for diameter estimation.

2.3 Statistics Computed

1. **Edge count** $|E|$ and **isolated nodes**.
2. **Degree metrics:** mean total degree, mean in-degree, mean out-degree.
3. **Diameter** (on the largest strongly-connected component).

Table 1: Global graph statistics

Metric	Symbol	Value
Nodes	$\lvert V \rvert$	6545
Edges	$\lvert E \rvert$	17816
Isolated nodes	—	790
Average degree	$\bar{d}$	5.444
Average in-degree	$\bar{d}_{in}$	2.7222
Average out-degree	$\bar{d}_{out}$	2.722
Diameter (largest SCC)	D	4

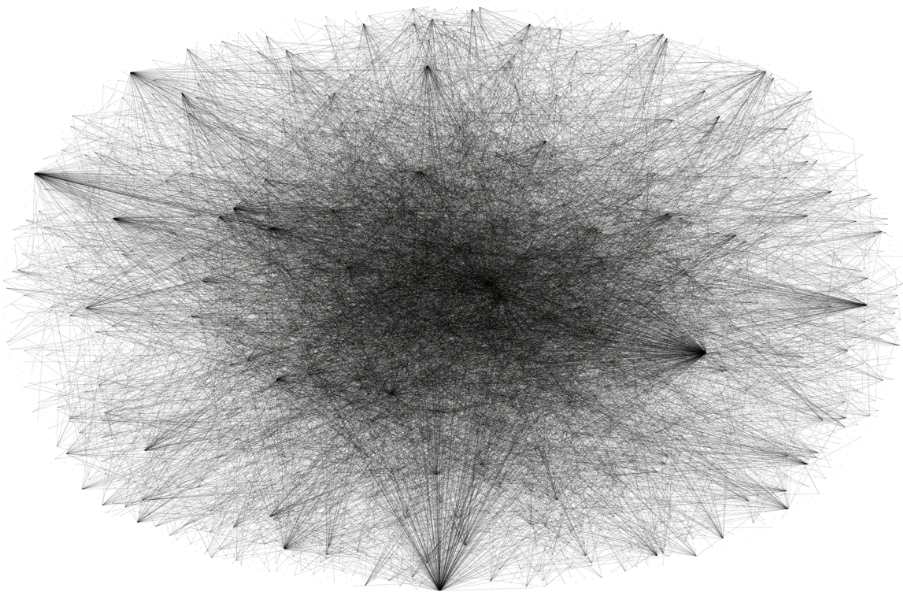


Figure 1: Snapshot of the citation sub-graph used for visual inspection.

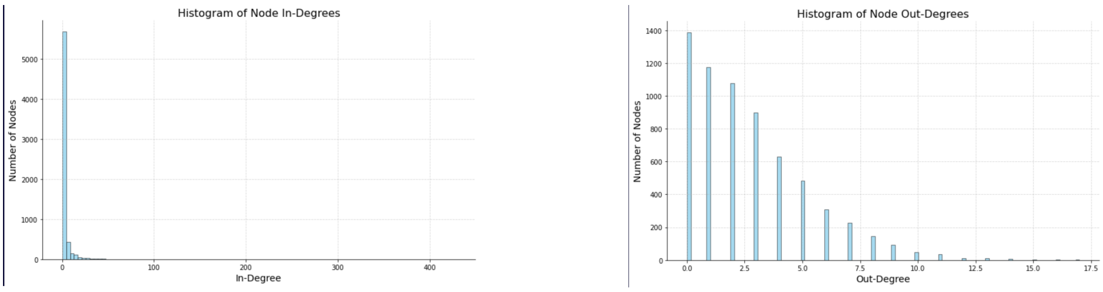
3 Results

4 Discussion

The heavy-tailed degree histograms (Figures 2a–3) confirm the expected power-law nature of academic citation networks, with a small core of highly cited papers and a long tail of sparsely cited work. The diameter of $D = 4$ suggests the "small-world" property still holds even for two decades of conference material.

5 Conclusion

We successfully transformed raw bibliographies into a directed citation graph and extracted fundamental network statistics. The pipeline is scalable to larger corpora (runtime dominated by bibliography parsing) and can serve as a foundation for downstream tasks such as link prediction or influence analysis.



(a) In-degree distribution

(b) Out-degree distribution

Figure 2: Histogram of directed degree counts (log-log scale).

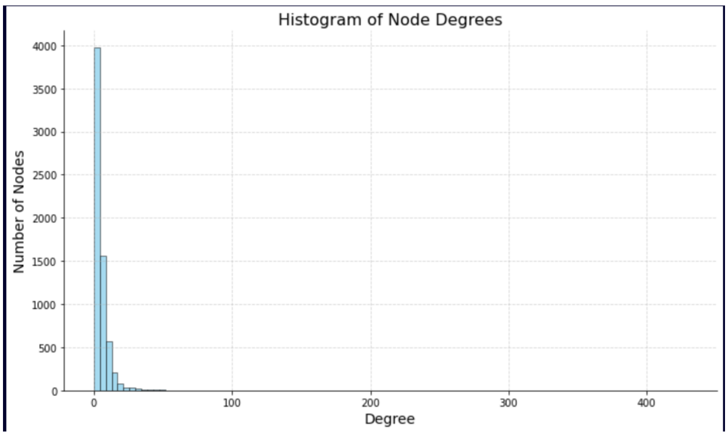


Figure 3: Total degree histogram for the whole graph.

6 Task 2: Machine-Learning Link Prediction

6.1 Problem Statement

Given a previously unseen paper, predict a ranked list of papers from the dataset that it is likely to cite. Formally, let q denote the query paper and $\mathcal{C}$ the set of all candidate papers. We learn a scoring function $s : (q, c) \mapsto R$ and return the top-K candidates $arg\,top^{K}_{c \in \mathcal{C}}\, s(q, c)$. Evaluation is carried out with **recall@K**; the actual value of K is hidden by the TAs. The file `evaluation.py` is invoked by `run_evaluation.py` and must print the top-K paper IDs for each query.

6.2 Approach Overview

1. **Citation graph construction** (Task 1 output).
2. **Text embeddings** with a pretrained Longformer (768-d vector per paper).
3. **GraphSAGE** (two layers) trained with contrastive loss to obtain 128-d node embeddings.
4. **Projection MLP** that maps Longformer vectors $R^{768} \rightarrow R^{128}$ so test papers live in the same space as GraphSAGE nodes.
5. **Link-prediction MLP** taking two 128-d embeddings and outputting the probability of a citation link.

Training pipeline Longformer embeddings $\rightarrow$ GraphSAGE message passing $\rightarrow$ contrastive loss on true vs. random citation pairs. The projection MLP is then fitted to minimise $f_{proj}(e_{LF})-e_{GS}{}_2^2$. Finally, the link-prediction MLP is trained on positive/negative edge pairs and can be mixed with cosine similarity (α-weighted).

Inference

a) Embed the query paper with Longformer.

b) Project to graph space via the trained projection MLP.

c) Score against every candidate paper's GraphSAGE embedding.

d) Rank by score and output the top-K IDs.

6.3 Future Work

- Replace Longformer with domain-specific language models (e.g. SciBERT, S2-Transformer).
- Train the whole pipeline end-to-end rather than step-wise.
- Filter candidates by publication year and simple heuristics (keyword overlap, venue match) to speed up inference and reduce false positives.